El Proceso Unificado y su evolución en EssUP

Maximiliano Gueroyte Muñiz

2007

UP & EssUP White Paper por Maximiliano Gueroyte se encuentra bajo una Licencia Creative Commons Atribución-NoComercial-SinDerivadas 3.0 Unported.

Resumen (Abstract)

Construir software es un proceso que da como resultado una versión particular del software que se está construyendo, la cual puede ser usada por una organización para trabajar. En las décadas pasadas se propusieron muchos modelos de desarrollo. Algunos métodos de desarrollo iterativo, tales como Evo[1], Microsoft Solution Framework[2], modelo en espiral[3] y UP. Este artículo se centrará en el método de desarrollo iterativo conocido como Proceso Unificado o UP y en su evolución como EssUP (Essential Unified Process) dónde, apoyándose en el Proceso Unificado, se toman las características esenciales de este para obtener una metodología igualmente valiosa pero mucho más simple, amigable y fácil de aprender.

El Proceso Unificado (Unified Process UP)

El Proceso Unificado fue desarrollado en conjunto con el lenguaje unificado de modelado (Unified Modeling Language UML).

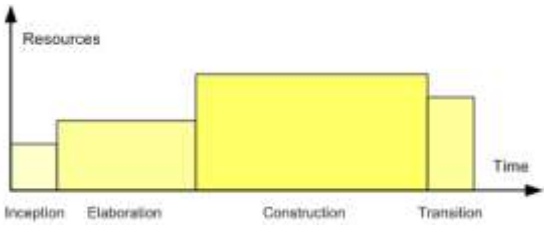
La razón por la que el Modelo de Proceso Unificado se discute en la definición de un marco para la arquitectura de software es claro: Las fases del Proceso Unificado y las tareas genéricas dan una definición precisa de los aspectos de la construcción de software. Por lo tanto, son una pauta para definir la dimensión "*Firmitas*"[4] de la arquitectura de software.

Características del Proceso Unificado

Iterativo e Incremental

El desarrollo iterativo es un método de construcción de productos cuyo ciclo de vida está compuesto por un conjunto de iteraciones, las cuales tienen como objetivo entregar versiones del software. Cada iteración se considera un subproyecto que genera productos de software y no sólo documentación, permitiendo al usuario tener puntos de verificación y control más rápidos e induciendo un proceso continuo de pruebas y de integración desde las primeras iteraciones.

El Proceso Unificado es un marco de desarrollo iterativo e incremental compuesto de cuatro fases denominadas:



- **Inicio (Inception):** Identificación de las necesidades del sistema de información. Esta es la fase más pequeña del proyecto, y que idealmente debería ser muy corta. Si la fase de inicio es larga entonces es generalmente una indicación de la excesiva especificación por adelantado, lo cual es contrario al espíritu del Proceso Unificado. Los

[1] Evo, "Evolutionary Project Management". Probablemente el primer método iterativo e Incremental de Desarrollo de Software, publicado en 1976. Creado por Tom Gilb.

[2] MSF se basa en experiencias y mejores prácticas de dentro y fuera de Microsoft, para aumentar las posibilidades de éxito de la ejecución de una solución, disminuyendo el número de personas en el equipo del proyecto, previniendo riesgos, y obteniendo resultados de calidad.

[3] El modelo de espiral combina las características del modelo de prototipos y el modelo de cascada. Es útil para proyectos complicados, grandes y costosos.

[4] El principio de "*Firmitas*" implica que la estructura debe ser sostenible y utilizable durante décadas.

siguientes son los objetivos típicos de la fase de inicio:

- Establecer una justificación u oportunidad de negocio para el proyecto.
- Establecer el alcance del proyecto y las condiciones de frontera.
- Resumir los casos de uso y los principales requisitos que impulsará el diseño.
- Esbozo de una o más arquitecturas candidatas.
- Identificar los riesgos.
- Preparar un calendario preliminar de proyecto y estimación de los gastos.

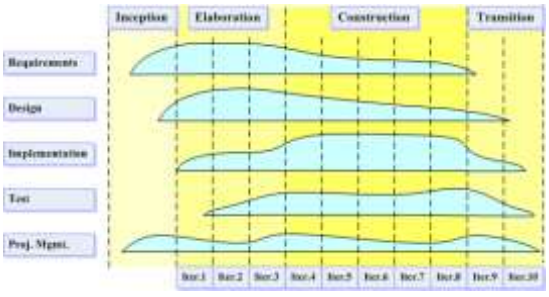
- **Elaboración (Elaboration):** Es la identificación y elección de los componentes necesarios del sistema. Durante la fase de elaboración se espera la captura de la mayoría de los requisitos del sistema. Sin embargo, el principal objetivo de la fase de elaboración es abordar los factores de riesgo conocidos y establecer y validar la arquitectura del sistema. La arquitectura es validada principalmente a través de la ejecución de una arquitectura Base. Se trata de una aplicación parcial del sistema que incluye los componentes más importantes. Se basa en una serie de pequeñas iteraciones. Al final de la fase de elaboración se debe tener estabilizada la arquitectura del sistema y demostrar que esta apoyará la funcionalidad clave, debe mostrar un comportamiento correcto en cuanto a rendimiento, escalabilidad y costo. La última etapa de elaboración es un plan de ejecución (incluyendo estimaciones de costo y calendario) para la fase de construcción.
- **Realización o Construcción (Construction):** Es la construcción y prueba de los distintos componentes de los sistemas, así como de las relaciones entre los componentes del sistema. La construcción es la fase más larga del proyecto. En esta fase el resto del sistema se basa en los conceptos establecidos en la fase de elaboración. La funcionalidad del sistema se implementa mediante una serie de breves iteraciones. Cada iteración da como resultado una versión ejecutable del software.
- **Transición (Transition):** Es la entrega del sistema en manos del usuario. La fase final del proyecto es de transición. Los comentarios recibidos de una primera versión (o versiones iniciales) pueden dar lugar a nuevas mejoras que se incorporarán mediante varias iteraciones de esta fase. La fase de transición también incluye la capacitación de los usuarios.

Cada una de estas fases es a su vez dividida en una serie de iteraciones (la de inicio sólo consta de varias iteraciones en proyectos grandes). Estas iteraciones ofrecen como resultado un incremento del producto desarrollado que añade o mejora las funcionalidades del sistema en desarrollo.

Cada una de estas iteraciones se divide a su vez en una serie de disciplinas que recuerdan a las definidas en el ciclo de vida clásico o en cascada:

- **Requisitos (Requirements):** Determinación de los requisitos para el sistema.
- **Análisis (Analysis):** Exploración del dominio del problema.
- **Diseño (Design):** Exploración del dominio de la solución, incluyendo la elaboración de soluciones alternativas para los problemas identificados en el análisis.
- **Implementación (Implementation):** Construcción de los componentes del sistema, y, en particular, la documentación de todos los componentes
- **Prueba (Test):** la prueba de los componentes y del sistema en su conjunto.

Aunque todas las iteraciones suelen incluir trabajo en casi todas las disciplinas, el grado de esfuerzo dentro de cada una de ellas varía a lo largo del proyecto.



Dirigido por los casos de uso

En el Proceso Unificado los casos de uso se utilizan para capturar los requisitos funcionales y para definir los contenidos de las iteraciones. La idea es que cada iteración tome un conjunto de casos de uso o escenarios y desarrolle todo el camino a través de las distintas disciplinas: diseño, implementación, prueba, etc.

Centrado en la arquitectura

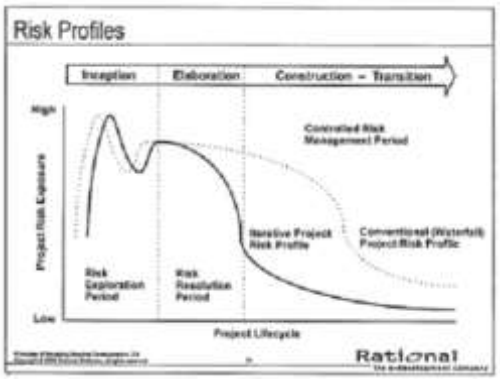
El Proceso Unificado asume que no existe un modelo único que cubra todos los aspectos del sistema. Por dicho motivo existen múltiples modelos y vistas que definen la arquitectura de software de un sistema. La analogía con la construcción de un edificio es clara, al construirlo existen diversos planos que incluyen los distintos servicios del mismo: electricidad, plomería, etc.

Enfocado en los riesgos

El Proceso Unificado requiere que el equipo del proyecto se centre en identificar los riesgos críticos en una etapa temprana del ciclo de vida. Los resultados de cada iteración, en especial los de la fase de Elaboración, deben ser seleccionados en un orden que asegure que los riesgos principales son considerados primero.

Cada proyecto tiene asociado intrínsecamente un conjunto de riesgos que requieren un plan de manejo claramente establecido, documentado y con una implementación eficaz. De esta manera se pretende evitar posibles retrasos en los tiempos de entrega, problemas de calidad en el producto o en el peor de los casos, que puedan afectar la culminación del proyecto. Estos procesos pueden ser tan complejos como la importancia del proyecto lo requiera. En las etapas iniciales se implementan las funcionalidades con mayor exposición al riesgo y las de mayor

complejidad, mejorando la posibilidad de éxito del proyecto.



5

Comparado con una metodología tradicional o "en cascada"[6], señalada en el gráfico con la línea punteada, en la fase inicial del proyecto, el nivel de exposición al riesgo en ambos modelos es casi igual, pero en las fases siguientes es completamente diferente para cada modelo. Este comportamiento se debe al período de exploración de riesgos del modelo iterativo (señalado en el gráfico con línea continua), donde se identifican, se priorizan y se define un plan de manejo para mitigarlos. Luego se procede a la fase de elaboración donde se implementan aquellos casos de uso que atacan los riesgos de más alta prioridad, lo cual se denomina período de resolución de riesgos. Al final de esta fase se debe tener definida la arquitectura del sistema, así como la infraestructura que lo soportará.

Proceso Unificado y Rational

El nombre Unified Process (en oposición a Rational Unified Process), es generalmente utilizado para describir el proceso genérico, incluyendo los elementos que son comunes a la mayoría de los refinamientos. El nombre Proceso Unificado también se utiliza para evitar posibles problemas de violación de copyright de Rational Unified Process y RUP que son marcas comerciales de IBM.

Refinamientos y Variaciones

Las mejoras en el Proceso Unificado varían según la forma categorizar las tareas o flujos de trabajo. El Rational Unified Process define nueve tareas: Modelado del negocio, requisitos, análisis, diseño, implementación, prueba, implantación, configuración y gestión del cambio, gestión de proyectos, y entorno. El Enterprise Unified Process extiende RUP a través de la adición de ocho tareas "corporativas". "Agile UP" como "Basic UP" simplifican RUP reduciendo el número de tareas.

Las mejoras también varían según el hincapié que se hace en diferentes artefactos[7]. Agile simplifica RUP mediante la simplificación de los flujos de trabajo y reduciendo el número previsto de artefactos.

Otras mejoras también varían en su especificación de lo que sucede después de la fase de transición. En el Rational Unified Process la fase de la transición suele ser seguida por una nueva fase de comienzo. En Enterprise Unified Process la fase de Transición por una fase de producción.

Es innumerable la cantidad de mejoras y variaciones del Proceso Unificado. Las organizaciones que utilizan el Proceso Unificado pueden incorporar sus propias

[5] Principles of Managing Iterative Development v.2.0 Rational Software Corporation.

[6] También llamado "Ciclo de vida básico" fue ingeniado por Winston Royce, y sugiere un enfoque sistemático o más bien secuencial del desarrollo de software que comienza en la especificación de requerimientos y progresa con el análisis, diseño, codificación, pruebas y mantenimiento.

[7] El término artefacto en relación con el desarrollo de software es asociado con métodos específicos de desarrollo

modificaciones y ampliaciones. La siguiente es una lista de algunas de las más conocidas mejoras y variaciones.

- Agile Unified Process (AUP), una ligera variación desarrollado por Scott W. Ambler
- Basic Unified Process (BUP), una ligera variación desarrollado por IBM y un precursor de OpenUP
- Enterprise Unified Process (EUP), una extensión del Rational Unified Process
- Essential Unified Process (EssUP), una ligera variación desarrollada por Ivar Jacobson [8]
- Open Unified Process (OpenUP), proceso de desarrollo de software de Eclipse Process Framework.
- Rational Unified Process (RUP), IBM / Proceso de desarrollo de Rational Software
- Rational Unified Process System Engineering (RUP - SE), una versión adaptada de RUP por Rational Software para Ingeniería de Sistemas

Essential Unified Process (EssUP) – Nueva vida para el Proceso Unificado

Como detallamos en el punto anterior, Ivar Jacobson desarrollo esta variación del Proceso Unificado de su coautoría.

EssUP se corresponde con el espíritu de "Lean Software"[9] que enfatiza en utilizar solo aquellas prácticas necesarias y que adaptan el proceso a las necesidades del proyecto. Usando un número reducido de categorías o tareas, EssUP proporciona varios procesos y artefactos que fácilmente describen, organizan y comunican.

[8] Ivar Hjalmar Jacobson (nacido en Suecia en 1939) Master en ingeniería eléctrica y Doctor en el Instituto Real de Tecnología "Construcciones Lingüísticas para Grandes Sistemas de Tiempo Real". Escribió varios libros. Inventó los Casos de Uso trabajando para Ericsson y junto a Grady Booch y James Rumbaugh crearon primero UML y luego RUP (comprado por IBM en el 2003).

[9] Mary and Tom Poppendieck son los autores de "Lean Software Development - An Agile Toolkit for Software Development Managers". Este libro presenta un set de 22 "herramientas".

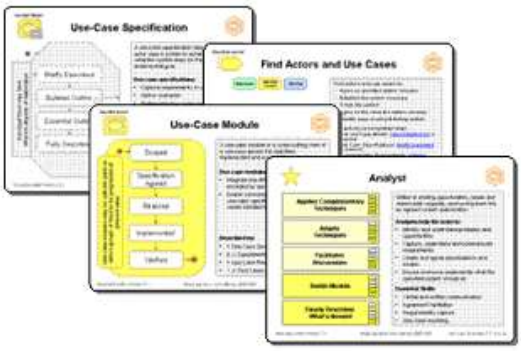
EssUP es el primero de una nueva generación de metodologías de desarrollo de software. Es ágil, iterativo, y liviano.

Enfocado en lo esencial

En esta metodología se quitó todo lo que no fuera esencial, y las prácticas que quedaron fueron mejoradas sustancialmente, agregando nuevas ideas que simplifican y fortalecen los nuevos procesos.

Presentado como un conjunto de tarjetas

Esta es la característica más innovadora de esta nueva versión del Proceso Unificado. Cada práctica se presenta como un conjunto de tarjetas de procesos que contienen los elementos necesarios para trabajar según el gusto de cada integrante del equipo de desarrollo. Incluyen tarjetas para la definición de competencias, de actividades, de artefactos, y patrones.



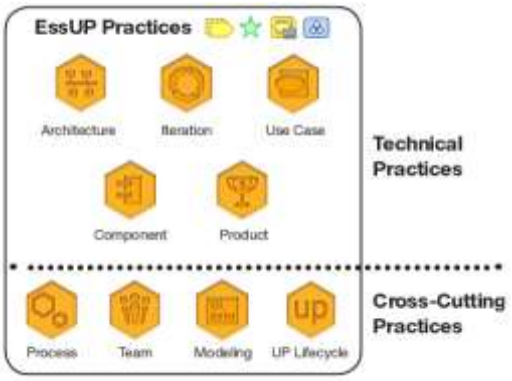
Las tarjetas pueden verse de forma electrónica o virtual, o bien tarjetas físicas que pueden ser manipuladas fácilmente para construir tareas, facilitar la adopción del método, ayudar el planeamiento de las iteraciones, o bien proveer una guía de referencia práctica mientras se trabaja.

Cada tarjeta tiene una guía que provee una mejor orientación a los usuarios de esta metodología. Estas guías son documentos cortos que muestran solo la información esencial que se necesita para usar la tarjeta.

Prácticas esenciales

Hay cinco prácticas técnicas (Technical Practices) para asegurar el desarrollo de un buen software y cuatro transversales (Cross-Cutting Practices)

para promover el trabajo en equipo y la mejora de los procesos:



Arquitectura: Captura la esencia del desarrollo centrado en la arquitectura asegurando que la arquitectura elegida para el proyecto es la correcta para el propósito.

Iteraciones: Permite adoptar fácilmente un enfoque iterativo para administrar y monitorear el proyecto, así como manejar los riesgos.

Casos de uso: Permite capturar los requerimientos de una forma ágil y usarlos para manejar el desarrollo y prueba de la solución.

Componentes: Permite desarrollar el software de forma simple y escalable explotando las propiedades de la orientación a objetos.

Procesos: Asegura la mejora continua de los procesos y ayuda a adoptar la forma de trabajo necesaria.

Equipo de trabajo: Captura la esencia de un equipo de trabajo manteniéndolo unido y ayudando a colaborar más eficientemente.

Modelado: Describe un acercamiento ágil al modelado, ayudando a adoptar un nivel de detalle apropiado, mejorando la comunicación y reduciendo el riesgo.

Ciclo de vida del Proceso Unificado: Tal como se vio anteriormente provee una serie de fases e hitos para ayudar a planear y seguir proyectos iterativos.

EssUP está incluido en la plataforma EssWork[10] y disponible para Eclipse, Visual Studio [11]y Web 2.0.

Metodología abierta y extensible

Antes, las organizaciones y fabricantes proveían una gran cantidad de información para comprender sus metodologías, en algunos casos duplicando conceptos y documentándolos en libros y "papers".

EssUP se enfoca en la esencia y provee referencias a trabajos fundamentales originales de la comunidad del software. Contrariamente a generar páginas y páginas de casos de uso, arquitectura o patrones, simplemente referencia los mejores "papers" o libros del mercado. Esto se realiza para evitar cualquier intento de aparecer como una fuente autoritaria y dejar en claro que los profesionales tienen la experiencia que es esencial para llevar a cabo el desarrollo de software.

Ivar Jacobson anunció que está buscando la mejor forma de crear una comunidad abierta EssUP, para compartir las mejores prácticas.

Conclusión

El Proceso Unificado es una metodología completa y bien documentada. Se usa como una fuente de ideas y herramientas ofreciendo una amplia cantidad de técnicas y prácticas. Sin embargo, a menos que haya un verdadero experto en el personal, es probable que no aumente significativamente la probabilidad de éxito. El proceso es complicado, difícil de aprender, y difícil de aplicar correctamente. Por el contrario, EssUP es mucho más simple, más

[10] EssWork consta de un entorno para la navegación, la composición, la construcción y la adaptación de las prácticas en procesos. No se trata de un proceso de desarrollo, o método. Es una manera de combinar las prácticas de muchas fuentes diferentes para producir formas de trabajo flexibles, coherentes y eficaces.

[11] El 5 de noviembre de 2007 se anunció que EssWork soporta Microsoft Visual Studio 2005 Team System y pronto Microsoft Visual Studio Team System 2008

flexible y extensible. Presenta una forma amigable de aprender la metodología fácilmente.

Referencias

1. Ivar Jacobson, Grady Booch y James Rumbaugh. The Unified Software Development Process. Rational Software Corporation. Addison-Wesley, 1999.
2. Rational Software Corporation. Principles of Managing Iterative Development v.2.0. 2001.
3. Robin Alberto Castro Gil. Paper Estructura básica del proceso unificadode desarrollo de software. Universidad Icesi, 2004.
4. Dave Thomas. Paper The Essential Unified Process (EssUP) - New Life for the Unified Process. Bedarra Research Labs, 2003.
5. Ivar Jacobson. Paper The Essential Unified Process – an introduction. Ivar Jacobson International, 2005.
6. Thomas Meloche. The Rational Unified Process (RUP), A Well Documented, Complete yet Complex Methodology. The Menlo Institute LLC, 2002.